Министерство науки и образования РФ

Федеральное государственное автономное образовательное

учреждение высшего профессионального образования

«Санкт-Петербургский государственный электротехнический

университет «ЛЭТИ» им. В. И. Ульянова (Ленина)»

(СПбГЭТУ «ЛЭТИ»)

Факультет компьютерных технологий и информатики

Кафедра вычислительной техники

Отчет по лабораторной работе №1

на тему:

“Структура программы на языке С++. Оператор ветвления”

по дисциплине “Программирование”

Вариант 5(2с).

Выполнил: Быков И. В.

Группа 6383

Принял: Самойленко В. П.

Санкт-Петербург  
2016 г.

**Цель**

Получение навыков использования операторов циклов с неизвестным количеством повторений. Ознакомление с особенностями вычислений с целыми числами.

**Краткие теоретические сведения**

1) Цикл *while* – используется, когда необходимое количество повторений тела цикла заранее неизвестно и зависит от некоторого условия.

Схема:

*while* (<условие>) {

операторы;

}

2) Целочисленные данные.

Используем в данной программе типы данных *int* и *unsigned int.*

*unsigned int*  хранит только положительные целые значения.

UINT\_MAX содержит максимальное значение для *unsigned int,* определен в *climits.*

3) Рекуррентные соотношения.

Последовательность чисел называется рекуррентной последовательностью m-ого порядка, если элемент выразить как функцию от m предыдущих элементов:

**Задание**

Для заданного целого вычислить , где - заданное число.

**Уточнение задания**

Требуется:

1. Организовать вычисления по схеме, приведенной в методичке. Использовать только целочисленные переменные.
2. Реализовать проверку вводимых данных.
3. Организовать вывод промежуточных данных.
4. Представить вывод рекуррентных соотношений.
5. Отслеживать переполнение и прерывать вывод при его обнаружении.
6. Определить диапазон значений входных данных, для которых программа выдает корректные выходные данные, для различных целочисленных типов – *unsigned short*, *unsigned int, unsigned long, unsigned long long.*

**Рекуррентное соотношение.**

Рекуррентное соотношение для суммы:

Найдем рекуррентное соотношение для

1. Найдем
2. Отсюда,

**Таблица экспериментальных данных.**

|  |  |
| --- | --- |
| Тестовые данные | Ожидаемые значения |
| ааа 12 | Введены нечисловые данные |
| -3 2 | Введено отрицательное число |
| 7 5 | Программа выведет промежуточный и окончательный результаты |
| 6 20 | Программа выведет промежуточне результаты и сообщит о переполнении |

**Текст программы с комментариями**

#include <stdio.h>

#include <iostream> //setlocale()

#include <cstdlib> //atoi()

#include <string.h> //strcmp()

#include <climits> //UINT\_MAX

void Function\_Find\_S(int , int);

int main()

{

setlocale(LC\_CTYPE , "rus"); //функция перекодирует программу на требуемый язык

char n\_str[256];

char k\_str[256];

int n = 0;

int k = 0;

bool flag = false;

printf("Введите n и k через пробел или enter:\n(Примечание: введенные числа должны быть целочисленными. В случае ввода числа с десятичной дробью в решении будет использована целая часть этого числа)\n");

while (!flag)

{

scanf("%s %s", n\_str, k\_str);

n = atoi(n\_str); // преобразует в int (можно использовать strtol для преобразования //в long int)

k = atoi(k\_str);

if ((strcmp(n\_str, "0") && (n == 0)) || (strcmp(k\_str, "0") && (k == 0))) // делаем //проверку, являются ли наши символами числами. При этом нужно помнить, что atof //преобразует и строку, и нуль в нуль

printf("Вы ввели нечисловые данные! Повторите ввод!\n"); // Т.е. нам нужно проверить, вводился ли в аргументы нуль сначала

else if ((n < 0) || (k <= 0))

printf("Число n должно быть неотрицательным, число k - положительным. Повторите ввод!\n");

else

flag = true;

}

Function\_Find\_S(n , k);

return 0;

}

void Function\_Find\_S(int arg\_n , int arg\_k)

{

unsigned int i = 0; // номер шага

unsigned int s = 1; // текущая сумма последовательности

unsigned int a = 1; // текущее значение а

bool IsOverFlow = false; // флаг переполнения

while ((i <= arg\_n) && !IsOverFlow)

{

printf("i = %d\ta = %d\t s = %d\n", i , a , s);

a \*= arg\_k; // находим к в i-ой степени

s += a; // находим i-ую сумму

i++;

if ((i <= arg\_n) && (((UINT\_MAX - s)/arg\_k) < a))

IsOverFlow = true;

}

if (!IsOverFlow)

printf("\nПри i = %d последнее слагаемое a = %d , итоговая сумма s = %d\n", i , a , s);

else

printf("\nОшибка! При i = %d произойдет переполнение.\n", i);

printf("Последний результат до переполнения: i = %d , слагаемое а = %d , сумма s = %d" , i - 1 , a / arg\_k , s - a);

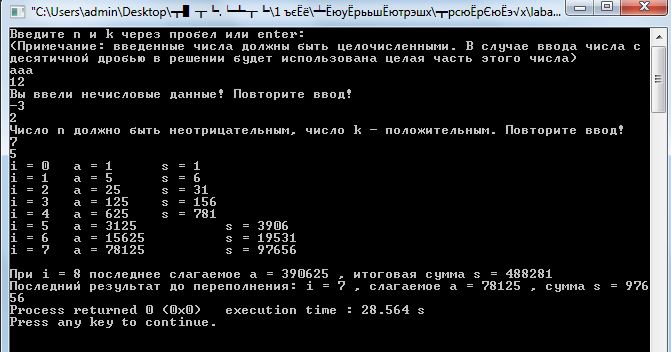
return;

}

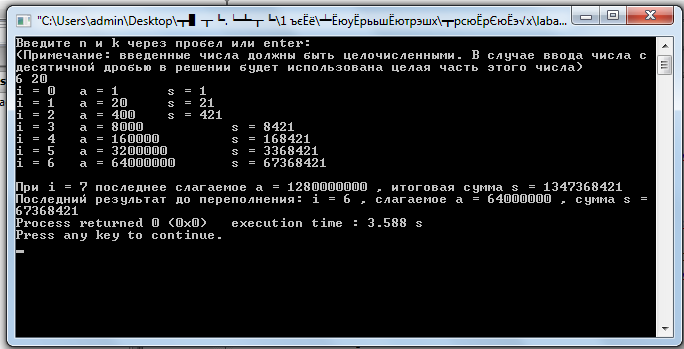
**Результаты решения задачи**

Введем сначала ошибочные данные (ааа , 12) и (-3 , 2) , затем введем (7 , 5)

Получим:



Введем (6 ; 20). Получим:



**Вывод**

При выполнении лабораторной работы были получены навыки использования операторов циклов с неизвестным количеством повторений. Также я ознакомился с особенностями вычислений с целыми числами. Получил дополнительный опыт в получении рекуррентных соотношений.